FORSCHUNGSZENTRUM JÜLICH GmbH
Zentralinstitut für Angewandte Mathematik
D-52425 Jülich, Tel. (02461) 61-6402

Interner Bericht

UNICORE
Das Fenster zum Supercomputer

Mathilde Romberg

FZJ-ZAM-IB-9908

Juni 1999

(letzte Änderung: 23.06.99)

Preprint: Proceedings of 14th Supercomputer Conference, Mannheim, June 10 - 12, 1999
(CD) ISBN Nr. 3-932178-08-4

UNICORE - Das Fenster zum Supercomputer

Mathilde Romberg
Forschungszentrum Jülich GmbH
Zentralinstitut für Angewandte Mathematik
D-52425 Jülich
e-mail: m.romberg@fz-juelich.de

Zusammenfassung: Höchstleistungsrechner sind für Forschung, Lehre und Industrie unverzichtbar geworden. Da jedoch Rechner verschiedener Hersteller zum Einsatz kommen, besteht ein erheblicher Lernaufwand, bevor ein neuer Rechner insbesondere über Netze genutzt werden kann. Benutzer, die für ihre Anwendungen Supercomputer benötigen, sind gezwungen, sich mit den rechenzentrums- und maschinenspezifischen Gegebenheiten vertraut zu machen. Dieses kann sehr zeitaufwendig werden, insbesondere wenn Rechnerkapazitäten in verschiedenen Zentren genutzt werden z.B. auch für heterogenes Rechnen oder Metacomputing. UNICORE (Uniformes Interface für Computer-Ressourcen) wurde entworfen, um den Aufwand für die Benutzer zu minimieren. Der Vortrag zeigt auf, wie der im UNICORE-Projekt entwickelte Prototyp durch einen einheitlichen und sicheren Zugang zu verteilten Supercomputer-Ressourcen Unterstützung gibt. Neben der Architektur wird insbesondere auf die entwickelte Oberfläche als Benutzerwerkzeug eingegangen.

1. Motivation

Warum ein Fenster zum Supercomputer? Betrachtet man die Situation der Rechnerbenutzer aus dem wissenschaftlichen Umfeld, die zur Lösung ihrer Probleme aus Physik, Chemie und anderen Bereichen Supercomputer-Rechenleistung benötigen, so zeigt sich, daß diese häufig mehr Rechenzeit benötigen als sie an einem Rechenzentrum für ihr Projekt erhalten. Für diese Benutzer ist es damit notwendig, ihre Anwendung an mehreren der Supercomputerzentren laufen zu lassen. Dieses wiederum bedeutet, daß sie sich um die system- und installationsspezifischen Gegebenheiten am jeweiligen Rechenzentrum kümmern müssen. Denn obwohl die meisten Betriebssysteme UNIX-Derivate sind und viele Batchsysteme NQS-Dialekte, haben die Befehle andere Defaults und Optionen. Zusätzlich sind die lokalen Voreinstellungen und Regelungen wie Sicherheitsmechanismen, Benutzerverwaltung, Datenhaltung und Jobscheduling zwischen den Zentren recht unterschiedlich. Gleichzeitig ist es wichtig, daß die vorhandenen Ressourcen an den Zentren effizient genutzt werden und nicht der hohe Lernaufwand Benutzer davon abhält, die für ihre Anwendung passende Architektur zu wählen. Desweiteren stehen auch nicht an allen Zentren alle von den Benutzern benötigten Anwendungspakete zur Verfügung, so daß eine einheitliche Infrastruktur für die Nutzung der Rechenzentren wichtig ist: Ein Werkzeug für einen „seamless access“, ein einheitliches Interface für den Zugang zu den Ressourcen ist notwendig.

2. Zielsetzung und Architektur

Das UNICORE-Projekt[1] hat zum Ziel, die oben beschriebenen Stoßkanten zwischen den Systemen und zwischen den Rechenzentren für den Benutzer unsichtbar zu machen und eine Software-Infrastruktur für einen einheitlichen, intuitiven und sicheren Zugang zu den Zentren zu schaffen. Die in dem Ziel enthaltenen Teilziele beinhalten im einzelnen:

- Der einheitliche Zugang impliziert zum einen die Definition einer systemunabhängigen Job-Beschreibung zur Formulierung von abstrakten Jobs. Zum anderen ist eine einheitliche UNICORE Benutzeridentifikation unabdingbar.
- Der intuitive Zugang wird erzielt durch eine ansprechende Benutzeroberfläche.
- Der sichere Zugang wird über eine tragfähige, auf Standards basierende Sicherheitsarchitektur erreicht.

Daneben darf UNICORE nur minimale Veränderungen der lokalen administrativen Verfahren wie z.B. Benutzerverwaltung und Scheduling erfordern, um eine leichte Integration in die Zentren zu ermöglichen. Weitere Ziele von UNICORE sind, existierende Techniken auszunutzen (z.B. das World Wide Web), um nicht das Rad überall neu zu erfinden, und Schnittstellen offen zu definieren, damit UNICORE keine Insel schafft, sondern offen für die Zusammenarbeit mit anderen ist. Das Projekt entwickelt einen Prototypen, der von den Supercom-

[1] Am vom BMBF geförderten UNICORE-Projekt beteiligt sind die KMU Genias Software GmbH und Pallas GmbH, die Forschungseinrichtungen und Universitäten Deutscher Wetterdienst Offenbach (+ ECMWF), Forschungszentrum Jülich, Rechenzentrum der Universität Stuttgart, Konrad Zuse Zentrum Berlin, Leibniz-Rechenzentrum München, Paderborn Center for Parallel Computing und Rechenzentrum der Universität Karlsruhe, das Fujitsu European Centre for Information Technologie und die Rechnerherstellerfirmen Hitachi, IBM, NEC, SGI, Siemens/Fujitsu, Sun. Weitere Informationen zum Projekt sind im Web verfügbar: http://www.fz-juelich.de/unicore

puterzentren genutzt wird und der zu einem Produkt weiterentwickelt werden soll, das auch von Hochschulen und Industrie eingesetzt wird.
Die im UNICORE Projekt zu realisierenden Funktionen sind das Erstellen und Überwachen von Batchanwendungen, das heterogene Rechnen in dem Sinn, daß einzelne Job-Schritte in einer festlegbaren Reihenfolge auf verschiedenen Systemen bearbeitet werden, und das transparente Bereitstellen der vom Benutzer spezifizierten Daten.

Im Projekt wurde die im folgenden kurz beschriebene drei-Ebenen-Architektur (siehe Abbildung 1) entwickelt: Sie besteht aus Benutzer-, UNICORE- und Systemebene. Die Kommunikation zwischen den Ebenen basiert zum einen auf dem https-Protokoll zwischen Benutzer- und UNICORE-Ebene, deren Verbindung über öffentlich Netze läuft und somit besonders geschützt werden muß. Zum anderen werden zwischen UNICORE- und System-Ebene für die Kommunikation auf die Installation abgestimmte TCP/IP-Dienste eingesetzt. Da die Daten nur lokal übermittelt werden, ist eine verschlüsselte Übertragung über https an dieser Stelle nicht notwendig. Das https-Protokoll ist Bestandteil der Sicherheitsarchitektur ([1],[2]) und sorgt zusammen mit den X.509-Zertifikaten für die gegenseitige Authentisierung der beteiligten Komponenten und für die sichere Kommunikation (siehe auch [4]). Über das X.509-Benutzerzertifikat ist gleichzeitig die einheitliche Benutzeridentifikation gegeben. Die zweite zentrale Komponente innerhalb von UNICORE ist die abstrakte Jobbeschreibung in der Form des Abstract Job Object ([[5]]).
Die Benutzerebene besteht aus dem Benutzerarbeitsplatz (UNIX-Workstation oder PC), auf dem der Web-Browser läuft und der Benutzer sein UNICORE X.509 Zertifikat vorhält. Dieses dient zur Authentisierung und ist gleichzeitig die einheitliche, zentrenübergreifende UNICORE-Benutzeridentifikation. Bei der Anwahl der Web-Adresse der UNICORE-Benutzeroberfläche weist sich zunächst der https-Server mit seinem Zertifikat gegenüber dem Benutzer aus bevor dieser sein Zertifikat an den Server weitergibt. Das Benutzerzertifikat liegt passwortverschlüsselt in der Datenbank des Web-Browsers auf Benutzerseite, so daß das Zertifikat nur von der berechtigten Person verwendet werden kann.
Der https-Server prüft das Benutzerzertifikat. Im nächsten Schritt wird der Job Preparation Agent (JPA) zur Erstellung und Submission eines Jobs oder der Job Monitor Controller (JMC) zur Job-Überwachung vom Server in den Web-Browser geladen. JPA und JMC sind signierte Java-Applets, da die Programme für den Benutzer sowohl auf lokale Daten seiner Workstation lesend und schreibend zugreifen müssen als auch eine Netzwerkverbindung zu einem beliebigen UNICORE-Server aufbauen müssen. Beides ist für nicht-signierte Applets verboten. Zusammen mit den Applets stehen die Ressourcen-Informationen der Zielsysteme zur Verfügung. Mit Hilfe der graphischen Benutzeroberfläche wird der UNICORE-Job erstellt und in der abstrakten Form (als AJO) an den Gateway-Teil im UNICORE-Server weitergeleitet. Die Aufträge werden vom Gateway dahingehend bearbeitet, daß die eindeutige Benutzerkennung aus dem UNICORE-Zertifikat anhand einer Benutzerdatenbank in die Benutzernummer am Zielsystem umgesetzt wird. Falls zusätzliche Authentisierungsinformation wie z.B. DCE-Tickets notwendig ist, kann das Gateway diese vom Benutzer einfordern und dem abstrakten Job beifügen. Der angepaßte Job wird an den Network Job Supervisor (NJS) weitergeleitet, der ihn von der abstrakten Form in einen für das Zielsystem geeigneten Job übersetzt und ihn dann submittiert. NJS sorgt auch dafür, daß die in einem UNICORE-Job enthaltenen Teil-Jobs, die für andere Zielrechner bestimmt sind, an das entsprechende System übermittelt werden. Bei Rechenzentren, die eine Firewall im Einsatz haben, liegt der Web-Server-Teil mit dem Gateway auf dem Firewall-Rechner und der NJS auf einem separaten System innerhalb der Firewall. Die dritte Ebene wird durch die Zielsysteme mit den zugehörigen Batchsubsystemen und Daten gebildet.

Die Abbildung 1 zeigt den Ausschnitt der UNICORE-Architektur bezogen auf ein Rechenzentrum. Jedes Zentrum, das UNICORE einsetzt, betreibt den UNICORE-Serverteil und natürlich Systeme, auf denen UNICORE-Jobs ausgeführt werden können. Abbildung 2 zeigt die Gesamtsicht der UNICORE-Architektur. Zwischen den UNICORE-Servern werden Teiljobs, Statusinformationen und Daten ausgetauscht. Dabei behält der erste Network Job Supervisor, bei dem ein kompletter UNICORE-Job ankommt, die Kontrolle über alle Teile und gibt dem Benutzer Zugriff auf die Ergebnisse. Details zur Architektur sind in [1],[2] beschreiben.

3. Benutzerfunktionen

Das Fenster, das dem Benutzer durch die oben beschriebene Architektur geboten wird, bietet eine einheitliche Benutzer-Schnittstelle zu (Super-) Computern. Unabhängig vom jeweiligen Zielsystem werden Ressourcenanforderungen und Kommandooptionen über eine graphische Oberfläche angegeben. Der JPA übersetzt die Angaben in den abstrakten Job (in das AJO) und später wird dieser vom NJS übersetzt in die konkreten Befehle und Optionsangaben für das vom Benutzer angewählte Zielsystem. Das X.509-Zertifikat des Benutzers dient als einheitliche UNICORE-Benutzeridentifikation. Der Prototyp unterstützt derzeit die Erstellung und die Überwachung von reinen Batch-Anwendungen. Die UNICORE Jobs können aus mehreren Teilen aufgebaut werden, die asynchron auf verschiedenen Rechnern an verschiedenen Rechenzentren ablaufen können. Der Benutzer kann die Teile mit sequentiellen Abhängigkeiten verbinden, über die auch die notwendigen Datentransfers spezifiziert werden.
Das in UNICORE verwendete Datenmodell beinhaltet, daß für jeden UNICORE-Job ein Job-Directory angelegt wird, welches das Basisdirectory während der Job-Ausführung auf dem Zielsystem ist. Benötigte Daten müssen jeweils in diesen Datenraum importiert bzw. aus ihm exportiert werden, wenn die Daten über das Job-Ende hin-

aus erhalten bleiben sollen. Zwischen Job-Schritten, die auf unterschiedlichen Zielsystemen ablaufen sollen, müssen die zu transferierenden Dateien bei der Definition der Abhängigkeit spezifiziert werden. UNICORE sorgt für den Datentransfer, der transparent für die Benutzer abläuft.

3.1. Erstellen von Jobs

Über den Job Preparation Agent (JPA), das graphische Benutzerinterface zur Erstellung und Submission von UNICORE Jobs, können derzeit Jobs aus den folgenden Elementen zusammengesetzt werden:

- Script Task, über die bereits bestehende Job-Scripts einbezogen werden können,
- Compile-Link-Run Task für das Einbinden neuer Applikationen und
- Job Gruppe, die rekursiv Teiljobs für ein anderes Zielsystem enthält.

Ein neuer UNICORE-Jobs wird im JPA über die Auswahl *New* im *File*-Menü erstellt. Mit dieser Funktion erzeugt der Benutzer den äußeren Rahmen für den UNICORE Job, der allgemeine, für den gesamten Job relevante Information wie

- das Zielsystem, das aus einer Liste der in UNICORE verfügbaren Zielsysteme ausgewählt wird,
- die email-Adresse des Benutzers, an die Nachrichten vom System geschickt werden sollen, und
- den Jobnamen[2]

enthält. Im GUI wird der Jobaufbau durch ein Baum-Symbol im linken Teil des Fensters angezeigt versehen mit dem gewählten Jobnamen. Die benötigten allgemeinen Informationen gibt der Benutzer in einem zusätzlich erscheinenden Fenster an. Der rechte Teil des GUI-Fensters ist zunächst leer. Über das *Job*-Menü kann der Benutzer die Elemente auswählen, die auf dieser Ebene des Jobs hinzugefügt werden sollen. Bei Auswahl einer Job Gruppe kann der Benutzer die Job-Attribute des UNICORE-Jobs für diesen Job-Teil ändern, z.B. ein anderes Zielsystem angeben. Wählt der Benutzer die Script- oder die Compile-Link-Run-Task aus, so werden die Komponenten-Symbole auf der linken Seite des GUI-Fensters mit Verbindung zum zugehörigen UNICORE-Job- oder Job Gruppen-Element hinzugefügt. Rechts erscheinen die Element-Symbole einer Job-Ebene zunächst ohne Struktur. Hier sind die Symbole verschiebbar und der Benutzer kann sie durch die Definition von Abhängigkeiten zwischen den Komponenten als gerichteten, azyklischen Graphen strukturieren.

Für die Elemente Script Task und Compile-Link-Run Task sind die benötigten Ressourcen zu spezifizieren. Zu den Ressourcen gehören:

- die Anzahl der Prozessoren (CPUs),
- der Hauptspeicherplatzbedarf,
- die CPU-Zeit und
- der Plattenplatzbedarf[2].

Hierbei werden jeweils die minimal und maximal möglichen Werte für das Zielsystem angezeigt, so daß der Benutzer keine Werte außerhalb des zulässigen Bereichs anfordern kann und auch sofort sieht, ob ein angewähltes Zielsystem die benötigten Ressourcen überhaupt anbietet.

Jedes Job-Komponenten-Symbol ist mit einem Farbfeld versehen, über das angezeigt wird, ob alle für die Submission notwendigen Informationen vorhanden und für das Zielsystem passend sind. Dabei steht rot für *noch nicht ok* und grün für *ok.* Nur wenn alle Komponenten und damit der ganze UNICORE Job grün gekennzeichnet sind, kann der Job submittiert werden; der entsprechende Befehl wird erst dann im Interface aktiviert und ausführbar. Das hat den Vorteil, daß nur Jobs, die von den Zielsystemen auch akzeptiert werden, submittiert werden können. Der Benutzer kann Fehler frühzeitig erkennen und beheben.

Die Abbildung 3 zeigt die graphische Oberfläche des JPA mit einem bereits erstellten Job. Auf der linken Seite sieht man die gesamte Job-Struktur, während die rechte Seite die Elemente mit ihren sequentiellen Abhängigkeiten auf der obersten Job-Ebene zeigt. Das Beispiel ist eine komplexe Anwendung aus dem Eurad-Projekt ([3], Europäisches Ausbreitungs- und Depositionsmodell), das sich mit der (Schad-) Stoff-Ausbreitung in der Atmosphäre befaßt. Die Anwendung besteht aus einer Reihe von Schritten, die zum Teil auf Vektorrechnern (CRAY T90) und zum Teil auf massiv-parallelen Systemen (CRAY T3E) abläuft. Bisher werden alle Schritte sukzessive gestartet und überprüft bevor der jeweils nächste Schritt gestartet wird. Über UNICORE kann in Zukunft die ganze Anwendung in einem Job zusammengefaßt werden und automatisch ablaufen und damit dem Benutzer Synchronisations- und Überwachungsarbeit abnehmen.

Ein Blick auf die Compile-Link-Run Task zeigt insbesondere die Mächtigkeit von UNICORE (siehe Abbildung 4, Abbildung 5). Diese Job-Komponente dient dazu, neue Anwendungen zielsystemunabhängig zu erstellen. Sie besteht intern aus drei Teilen, nämlich dem Compile-, dem Link- und dem Run-Teil, von denen jedes einzeln, beliebige Paare oder alle drei gemeinsam zum Jobaufbau benutzt werden können. Die Auswahl dazu ist oben im rechten Teil des graphischen Interfaces für die CLR-Task in der Form von Checkboxes gegeben. Wählt man im linken Teil des GUI eine CLR-Komponente aus, erscheint auf der rechten Seite des Fensters das Eingabe-Fenster für die Compile Task (siehe Abbildung 4). Der Eingabe-Bereich ist in vier Zonen geteilt: die oben genannten

[2] Über Preferences können Standardeinstellungen für diese Werte gesetzt werden.

Checkboxes, ein allgemeiner Teil mit Namen-, Ressourcen-, Kontextangaben, ..., ein Bereich für Input-Daten wie Source-Dateien und Compiler-Optionen und ein Bereich für Output-Daten. Wesentliche Bereiche für abstrakte, systemunabhängige Job-Definitionen, sind die *Resources*, die *Execution contexts...*, *Preprocess options* und die Compiler-Optionen wie *optimisation level, check argumnets, check bounds, listing level,* und andere. Da gibt der Benutzer zum Beispiel für die Anzahl der benötigten CPUs für seinen parallelen Job 12 an, was im AJO zum NJS transportiert wird und dort für ein Zielsystem mit Batchsubsystem CRAY NQS in die qsub-Option *–l mpp_p=12* übersetzt wird und für ein Zielsystem mit Batchsubsystem Codine in die qsub-Option *–pe <parallel_environment> 12*. Die Art des parallelen Programmiermodells (z.B. MPI) wählt der Benuzter bei den *Execution contexts...* aus. Das wird im obigen Beispiel für CRAY NQS nicht als qsub-Option übersetzt, sondern bei den Bibliotheken und z.B. bei der Programmausführung über mpprun berücksichtigt, während es bei Codine auch in die qsub-Option *–pe mpi 12* übersetzt wird. Analog werden die ausgewählten Compiler-Optionen auf das jeweilige Zielsystem abgebildet – völlig transparent für den Benutzer, egal welches Zielsystem er ausgewählt hat, seine Angaben im JPA sind immer dieselben. So wird die Benutzerangabe *Optimisation level agressive* aus dem Interface bzw. dem AJO vom NJS für den Fortran Compiler auf einer CRAY T3E in die f90 Option *–O3,unroll2* übersetzt, für eine NEC SX-4 in die f90 Optionen *–Wf,-dir par –P auto –C hopt –pi –Wf, -pvctl,fullmsg,vwork=stack,noassume,loopcht=10000000* .
Das GUI für den Run-Teil (siehe Abbildung 5) ist genauso gegliedert wie das für Compile; man kommt zu der Oberfläche, indem man den *Run*-Knopf in der Kopfzeile anschaltet. Dateien, die als Input-Daten angegeben werden, werden in das UNICORE-Job-Directory importiert, Output-Daten werden entsprechend exportiert. Das auszuführende Programm (*Executable*) kommt entweder aus dem Link-Schritt vorher und muß damit nicht angegeben werden, oder wird explizit angegeben und, wenn nötig, importiert.

Sehr wichtig für die Akzeptanz von UNICORE ist es, daß Benutzer auch bereits bestehende Job-Scripts über UNICORE abschicken können. Für diesen Zweck gibt es die Script Task Komponente (siehe Abbildung 6). Neben Jobnamen und Ressourcen-Anforderung gibt der Benutzer das Job-Script an, das entweder lokal auf seiner Workstation liegt oder von einem anderen System importiert werden soll. Das Script kann in dem GUI noch angepaßt und lokal abgespeichert werden.

Neben dem Erstellen eines neuen Jobs können auch „alte“, d.h. früher bereits erstellte und abgespeicherte UNICORE Jobs wieder in den JPA geladen werden (über die Auswahl *Load* im *File*-Menü). Diese Jobs können modifiziert werden indem Komponenten gelöscht, hinzugefügt oder abgeändert werden. Zum Beispiel können auch Resourcenanforderungen geändert werden: Ein Benutzer hat bisher Programmentwicklung gemacht und nur kurze Testläufe benötigt. Nun will er für den Produktionslauf die CPU-Zeit und/oder die Anzahl der CPUs heraufsetzen. Dieses kann der Benutzer über das *Resources* Panel der entsprechenden Jobkomponente angeben. Soll der UNICORE Job auf einem anderen Zielsystem ausgeführt werden, wählt der Benutzer auf der betroffenen Job-Ebene die Auswahl *Properties* an (unter dem rechten Teilfenster) und selektiert ein anderes Zielsystem aus der angebotenen Liste der verfügbaren UNICORE-Zielsysteme. UNICORE zeigt dem Benutzer alle Ressourcen-Angaben, die durch die Änderung des Zielsystems nun gegebenenfalls nicht mehr korrekt sind, d.h. unpassend für das Zielsystem sind. Damit hat der Benutzer sofort die Möglichkeit die Angaben der neuen Auswahl anzupassen oder das Zielsystem als nicht geeignet für die Anwendung zu verwerfen und eine andere Lösung zu suchen.

3.2. Überwachen von Jobs

Der Job Monitor Controller (JMC) ist der Teil des UNICORE-Benutzerinterfaces, über das der Status der UNICORE Jobs überwacht wird. Es ist analog zum JPA aufgebaut und zeigt im linken Teilfenster die vom Benutzer an UNICORE abgeschickten und noch aktiven Jobs. Dabei werden hier im Gegensatz zum JPA mehrere Job-Bäume angezeigt und nicht nur einer. Über Farben wird der jeweilige Jobstatus des UNICORE-Jobs und seiner Komponenten verdeutlicht. Grün bedeutet dabei *erfolgreich beendet*, rot steht für *mit Fehler beendet* bzw. *killed*, gelb zeigt *executing* an, blau *queued* und grau bedeutet *undefined*. Im Fall von gelb, ein Job wird aus UNICORE Sicht gerade ausgeführt, wird der Farbbalken zweigeteilt und in der zweiten Hälfte wird der Status aus Sicht des Batchsubsystems auf dem Zielrechner gezeigt: blau für *queued* und gelb für *running*. Im rechten Teilfenster ist die Jobstruktur der auf der linken Seite aus einem Job ausgewählten Ebene zu sehen mit den definierten Abhängigkeiten. Darüber wird klarer, warum z.B. eine Komponente des Jobs noch *queued* ist, wenn ihre Vorgängerin im Abhängigkeitsgraphen noch nicht beendet wurde.
Als Beispiel ist in Abbildung 7 der detaillierte Status einer Compile-Link-Run Task angegeben. Auf dieser Ebene erhält der Benutzer zur Zeit noch für den auf dem Zielsystem ausgeführten Job die Informationen aus der Standard-Ausgabe- und der Standard-Fehler-Datei, die er bei Bedarf abspeichern kann.
Neben der Überwachung des Jobstatus kann der Benutzer UNICORE Jobs löschen, wobei dann automatisch alle zugehörigen Jobkomponenten auf den Zielsystemen gelöscht werden. Je nach Anforderung bleiben die Standard-Dateien erhalten oder werden auch sofort gelöscht.

4. Ausblick

Der bisher im UNICORE-Projekt entwickelte Prototyp zeigt, daß ein einheitlicher, intuitiver und vor allem sicherer Zugang zu verteilten Rechnerressourcen machbar ist. Das Fenster zum Supercomputer erlaubt den Benutzern die für ihre Anwendungen geeigneten Rechnerarchitekturen und Rechner zu benutzen ohne jeweils erheblichen Aufwand in die Anpassung an die Gegebenheiten des Zielsystems und des Rechenzentrums zu investieren.
Über das Projekt hinaus ist geplant, UNICORE zu erweitern und im Projekt explizit ausgeklammerte Bereiche, für die von Anwenderseite jedoch sehr großes Interesse besteht, zu integrieren. Zu den möglichen Erweiterungen zählen das Metacomputing auf Anwendungsebene, die Unterstützung von Anwendungspaketen mit speziell auf sie zugeschnittenen JPA-Elementen sowie die interaktive Steuerung von Anwendungen.

5. Referenzen

[1] Jim Almond, Seamless Computing: Einheitlicher Zugriff auf Supercomputer über das Web, FOKUS Praxis Information und Kommunikation, Band 16, Hans-Werner Meuer (Hrsg.), Supercomputer 1998, K.G.Saur Verlag München

[2] Jim Almond and Dave Snelling, UNICORE: Secure and Uniform Access to Distributed Resources via the World Wide Web, A White Paper, October 1998, http://www.fz-juelich.de/unicore/whitepaper.ps.gz

[3] Das EURAD Projekt an der Universität zu Köln, http://www.uni-koeln.de/math-nat-fak/geomet/eurad/index.html

[4] Jalal Feghhi, Jalil Feghhi, and Peter Williams, Digital Certificates, Addison Wesley,1998

[5] Dave Snelling, The Abstract Job Object: An Open Framework for Seamless Computing, Vortrag beim 1.DATORR-Treffen, 8.10.98, Argonne National Laboratory, http://www.fz-juelich.de/unicore/DARR.ps.gz

6. Abbildungen

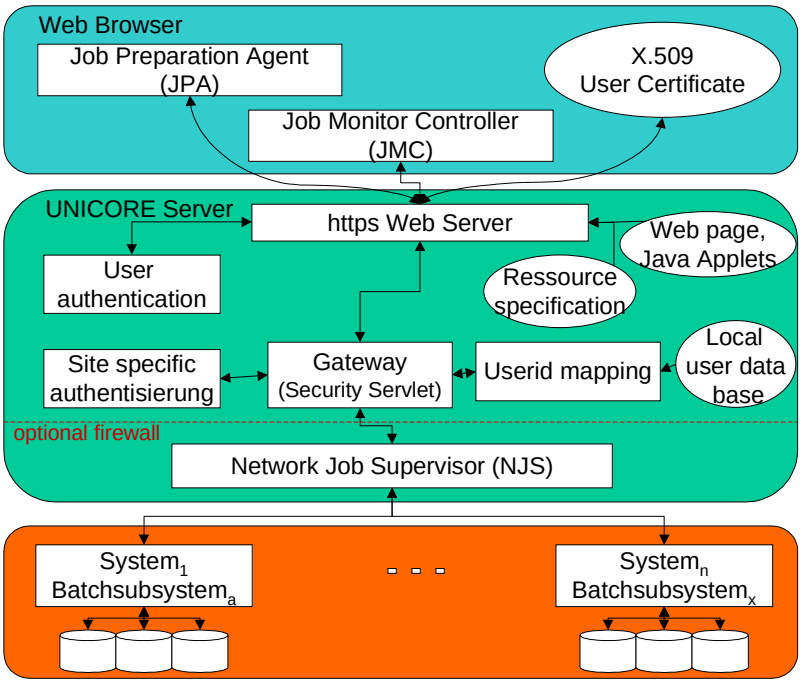

Abbildung 1: Die UNICORE Architektur (Detail-Ansicht)

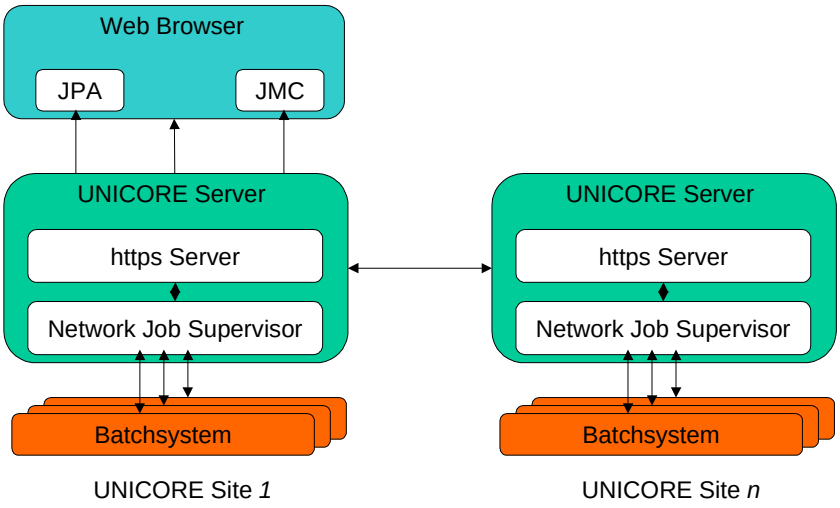

Abbildung 2: Die UNICORE Architektur (Gesamtsicht)

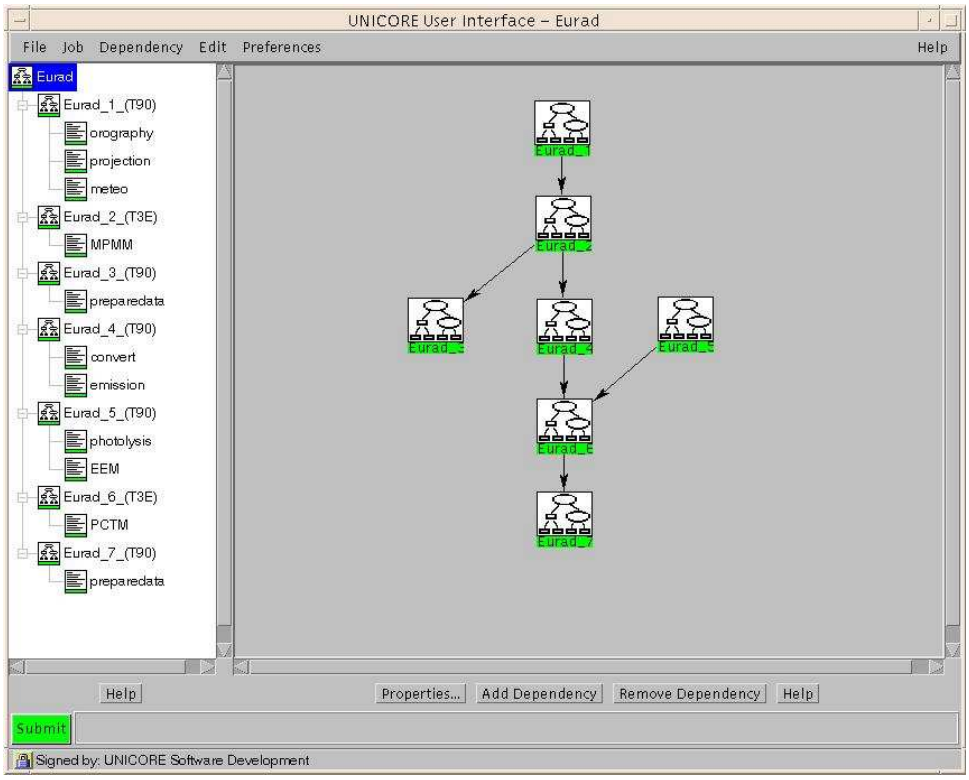

Abbildung 3: UNICORE-Job für das Eurad Modell

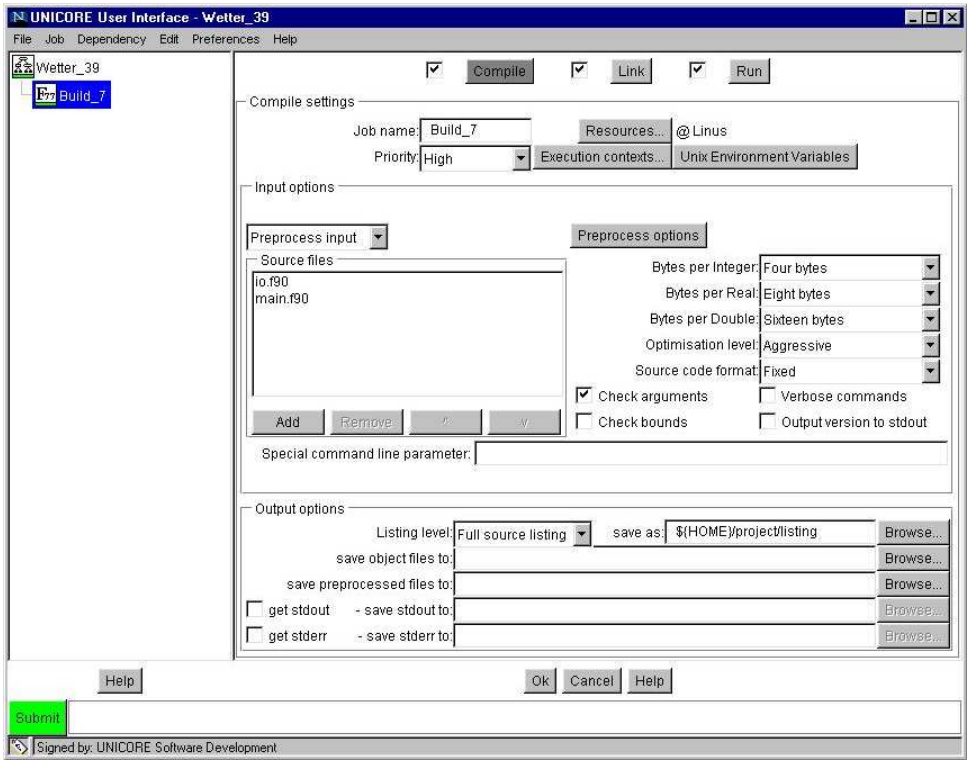

Abbildung 4: Beispiel für das Compile GUI

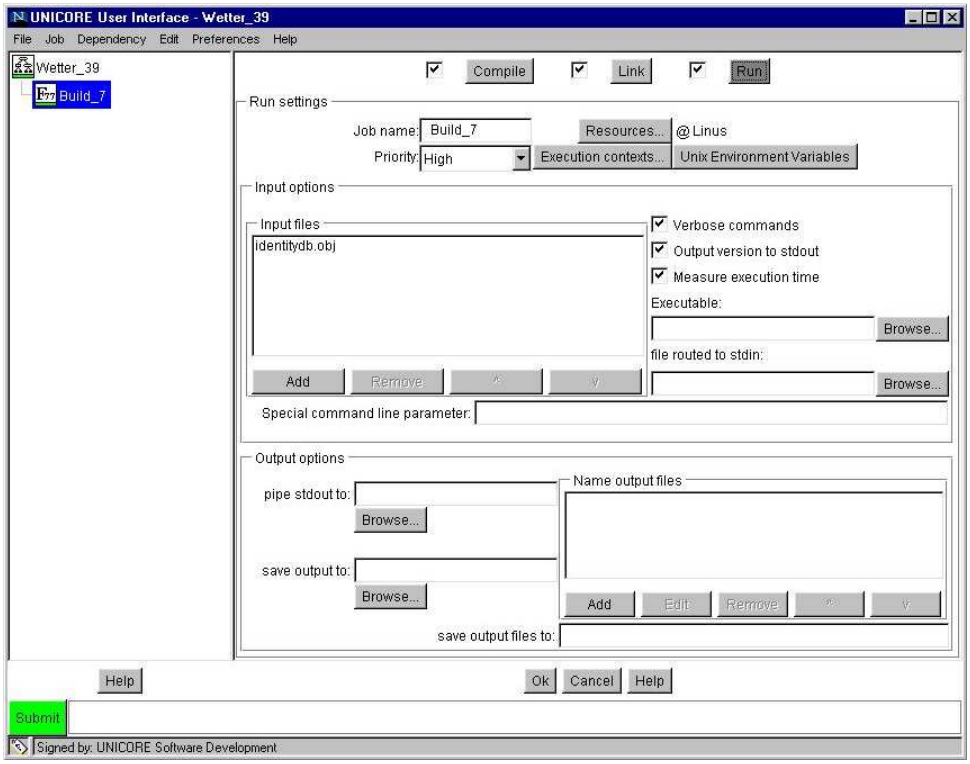

Abbildung 5: Beispiel für das Run GUI

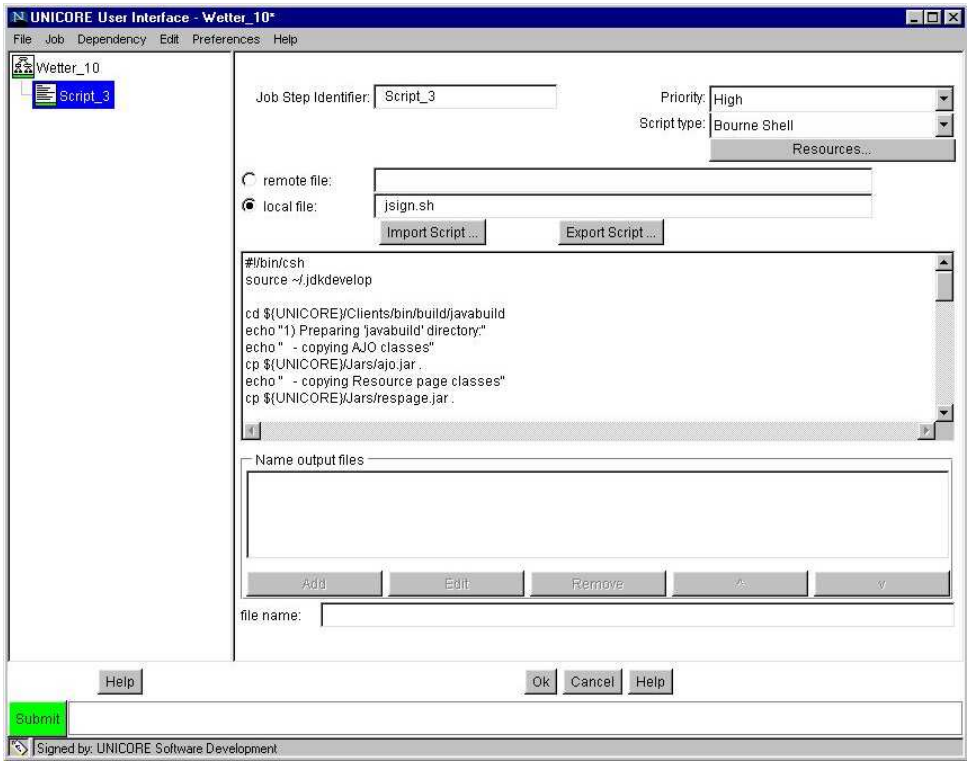

Abbildung 6: Beispiel für das Script Task GUI

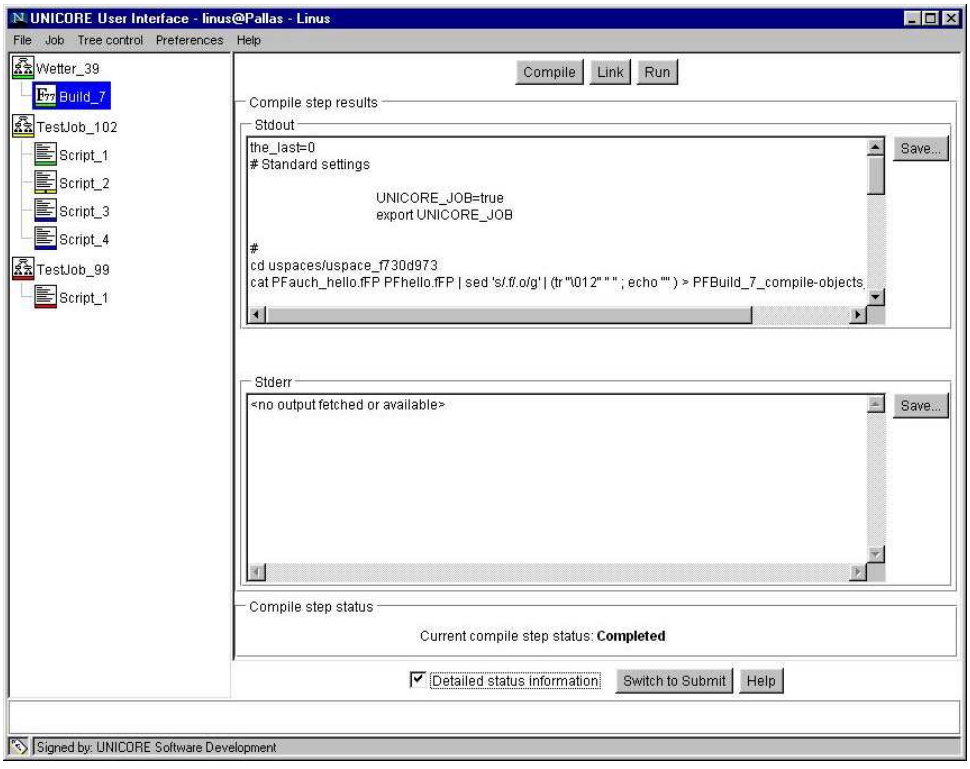

Abbildung 7: Beispiel für die Job-Status Anzeige